WHAT IS CLAIMED IS:

1. A method of buffer management and task scheduling for two-dimensional data transforming, comprising the steps of:

(a) reading out old data in a block-by-block pattern and immediately writing in new data in a line-by-line pattern using a first mapping scheme; and

(b) reading out a following old data in a block-by-block pattern and immediately writing in a following new data in a line-by-line pattern using a second mapping scheme.

2. The method of claim 1, wherein an initialization step prior to step (a) is performed so as to set up a write logic address and a read logic address to zero.

3. The method of claim 2, wherein the write logic address is incremented by 1 per write and the read logic address is incremented by 1 per read.

4. The method of claim 1, wherein the data transforming is a two-dimensional Discrete Cosine Transform.

5. The method of claim 1, wherein the first and second mapping scheme translate logical addresses to physical addresses.

6. A method of buffer management and task scheduling for two-dimensional data transforming, which transforms data in a buffer sequentially by blocks, wherein a block comprises a specific size having plurality of row portions and a plurality of column portions, comprising the steps of:

(a) reading out old data in a block-by-block pattern and immediately writing in new data in a line-by-line pattern in the block using a first mapping scheme;

(b) moving the block to transform another column portion; and

(c) reading out old data in a block-by-block pattern and immediately writing in new data in a line-by-line pattern in the block using a second mapping scheme.

7. The method of claim 6, wherein an initialization step prior to step (a) is performed so as to set up a write logic address and a read logic address to zero.

8. The method of claim 7, wherein the initialization step comprises completely filling the block with data using the second mapping scheme.

9. The method of claim 8, wherein the write logic address is incremented by 1 per buffer write and the read logic address is incremented by 1 per buffer read.

10. The method of claim 6, wherein step (d) comprises moving the block to transform another column portion.

11. The method of claim 10, wherein step (a) to step (d) are repeated until the whole data in the buffer is transformed.

12. The method of claim 6, wherein the data transforming is a two-dimensional Discrete Cosine Transform.

13. The method of claim 6, wherein the buffer is a pre-buffer.

14. The method of claim 6, wherein the buffer is a staging buffer.

15. The method of claim 6, wherein the block has an 8x8 block size.

16. The method of claim 6, wherein the data is an image comprising 512 pixels of columns and 8 lines of rows.

17. The method of claim 16, wherein each pixel has a 9-bit pixel address which is separated to a 3-bit dot address, 3-bit block address and a 3-bit sector address and wherein each line is a 3-bit line address.

18. The method of claim 6, wherein the first and second mapping scheme translate logical addresses to physical addresses.

19. The method of claim 18, wherein the physical address in the first mapping scheme is equivalent to {sector address, line address, block address, dot address}.

20. The method of claim 18, wherein the physical address in the second mapping scheme is equivalent to {sector address, block address, line address, dot address}.

21. A method of buffer management and task scheduling for two-dimensional data transforming, which transforms data in a buffer sequentially by blocks, wherein a block comprises a specific size having plurality of row portions and a plurality of column portions, comprising the steps of:

(a) initializing a write logic address and a read logic address to zero;

(b) reading out old data in a block-by-block pattern and immediately writing in new data in a line-by-line pattern in the block using a first mapping scheme, wherein the write logic address and the read logic address is incremented by 1;

(c) moving the block to transform another column portion; and

(d) reading out old data in a block-by-block pattern and immediately writing in new data in a line-by-line pattern in the block using a second mapping scheme, wherein the write logic address and the read logic address is incremented by 1.

22. The method of claim 21, wherein the step (a) comprises completely filling the block with data using the second mapping scheme and incrementing the write logic address by 1.

23. The method of claim 21, wherein step (e) comprises moving the block to transform another column portion.

24. The method of claim 23, wherein step (b) to step (e) are repeated until the whole data in the buffer is transformed.

25. The method of claim 21, wherein the data transforming is a two-dimensional Discrete Cosine Transform.

26. The method of claim 21, wherein the buffer is a pre-buffer.

27. The method of claim 21, wherein the buffer is a staging buffer.

28. The method of claim 21, wherein the block has an 8x8 block size.

29. The method of claim 21, wherein the data is an image comprising 512 pixels of columns and 8 lines of rows.

30. The method of claim 29, wherein each pixel has a 9-bit pixel address which is separated to a 3-bit dot address, 3-bit block address and a 3-bit sector address and wherein each line is a 3-bit line address.

31. The method of claim 21, wherein the first and second mapping scheme translate logical addresses to physical addresses.

32. The method of claim 31, wherein the physical address in the first mapping scheme is equivalent to {sector address, line address, block address, dot address}.

33. The method of claim 31, wherein the physical address in the second mapping scheme is equivalent to {sector address, block address, line address, dot address}.